2007-11-09

High-assurance system support through 3-D integration

Huffmire, Theodore

Monterey, California. Naval Postgraduate School

NPS-CS-07-016

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

High-Assurance System Support through 3-D Integration

by

Theodore Huffmire, Tim Levin, Cynthia Irvine, Thuy Nguyen,
Jonathan Valamehr, Ryan Kastner, and Tim Sherwood

9 November 2007

Approved for public release; distribution is unlimited.

Prepared for the National Science Foundation

This page left intentionally blank

NAVAL POSTGRADUATE SCHOOL
Monterey, California 93943-5000

Daniel T. Oliver
President

Leonard A. Ferrari
Executive Vice President and
Provost

This report was prepared for and funded by: The National Science Foundation

This report was prepared by:

_____
Timothy E. Levin
Research Associate Professor

Reviewed by:

_____
Peter J. Denning
Department of Computer Science

Released by:

_____
Dan C. Boger
Interim Vice President and
Dean of Research

This page left intentionally blank

REPORT DOCUMENTATION PAGE

Form approved

OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	9 November 2007	Research; 11/9/06 – 11/9/07

4. TITLE AND SUBTITLE

High-Assurance System Support through 3-D Integration

5. FUNDING

Grant number: CNS-0524707

6. AUTHOR(S)

Theodore Huffmire, Timothy Levin, Cynthia Irvine, Thuy Nguyen,
Jonathan Valamehr, Ryan Kastner, and Timothy Sherwood

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Center for Information Systems Security Studies and Research (NPS CISR)
1411 Cunningham Rd., Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

NPS-CS-07-016

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Science Foundation, 4201 Wilson Blvd. 1175 N. ArlingtonVA22230

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

Not applicable

11. SUPPLEMENTARY NOTES

The views expressed in this report are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words.)

While hardware resources, in the form of both transistors and full microprocessor cores, are now fairly abundant, economic factors continue to prevent the integration into commodity parts of specialized hardware mechanisms required for secure processing. Multi-core processors, due to their wide adoption, impressive performance, and low cost, are very attractive platforms for computation. Unfortunately, highly secure processing of sensitive information on such platforms is extremely difficult to achieve due to extensive resource sharing and the lack of strong security primitives. In this paper we propose that commodity integrated circuits, with some very minor modifications, could be enhanced with a separate silicon layer used to enforce strong isolation, reference monitoring, and other useful security properties. A separate layer, stacked using 3-D integration, allows us to decouple the function and economics of high assurance policy enforcement mechanisms from the underlying high-performance computing hardware. We describe 3-D integration, how the host layer may be modified, and as our working example, we show how the problem of cache-based side channels can be addressed by re-routing signals from the computation layer through a cache manager in the control layer.

14. SUBJECT TERMS

Integrated Circuits, microprocessors, security and protection, reference monitor, high assurance, multilevel security

15. NUMBER OF PAGES

26

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

This page left intentionally blank

NPS-CS-07-016

RCsec Technical Report

High-Assurance System Support Through 3-D Integration

Theodore D. Huffmire, Timothy E. Levin, Cynthia E. Irvine,
Thuy D. Nguyen, Jonathan D. Valamehr,
Ryan Kastner and Tim Sherwood

9 November 2007

This page left intentionally blank

This material is based upon work supported by the National Science Foundation under Grant No. CNS-0524707. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Author Affiliations

Naval Postgraduate School:

Theodore Huffmire, Cynthia Irvine, Timothy Levin and Thuy Nguyen

Center for Information Systems Security Studies and Research
Computer Science Department
Naval Postgraduate School
Monterey, California 93943

University of California at Santa Barbara:

Jonathan Valamehr and Tim Sherwood

Department of Computer Science
University of California, Santa Barbara
Santa Barbara, CA 93106-5110

University of California at San Diego:

Ryan Kastner

Department of Computer Science and Engineering
University of California, San Diego
9500 Gilman Drive, Mail Code 0404
La Jolla, CA 92093-0404

This page left intentionally blank

High-Assurance System Support through 3-D Integration

Abstract

While hardware resources, in the form of both transistors and full microprocessor cores, are now fairly abundant, economic factors continue to prevent the integration into commodity parts of specialized hardware mechanisms required for secure processing. Multi-core processors, due to their wide adoption, impressive performance, and low cost, are very attractive platforms for computation. Unfortunately, highly secure processing of sensitive information on such platforms is extremely difficult to achieve due to extensive resource sharing and the lack of strong security primitives. In this paper we propose that commodity integrated circuits, with some very minor modifications, could be enhanced with a separate silicon layer used to enforce strong isolation, reference monitoring, and other useful security properties. A separate layer, stacked using 3-D integration, allows us to decouple the function and economics of high assurance policy enforcement mechanisms from the underlying high-performance computing hardware. We describe 3-D integration, how the host layer may be modified, and as our working example, we show how the problem of cache-based side channels can be addressed by re-routing signals from the computation layer through a cache manager in the control layer.

1 Introduction

Development of high-assurance computing systems is often caught between the competing pressures to provide complete and precise security policy enforcement and to exploit the abundance of performance and resources associated with commodity products and parts. High-assurance systems, such as those based on micro-kernels and virtual machine monitors built using off-the-shelf components, find themselves constantly battling the consequences of commercial pressures to rapidly increase functionality at the expense of separation, isolation, and protection. Further exacerbating this problem is the fact that, while hardware resources are now fairly abundant, due to market forces, security functionality is often not considered at the platform ISA or micro-architecture levels, thus creating exploitable features.

Market pressures drive manufacturers to release products as soon as the required functionality can be provided, leaving security considerations to the next release. Even when the hardware industry incorporates significant security enhancements, integrating these mechanisms into a complex design presents many practical and theoretical problems, driving up the costs and driving out the release schedule. Furthermore, security assurance processes, such as formal analysis of the security mechanisms, are far more difficult when the mechanisms are tangled into the product's functionality.

To divide and conquer these problems, we propose to disentangle the security mechanisms from the design, consolidating them onto a security overlay, *literally a separate layer* of circuitry that is stacked on top of a commodity integrated circuit. The security mechanisms that reside in this overlay can then be connected to the underlying chip with any number of die-stacking technologies, yet can be left unattached to enable the manufacturer to continue to sell the un-enhanced product at a lower cost.

Attaching multiple layers of silicon together in 3-D stacks is a new yet already marketed technology [46], which is being explored by most of the major microprocessor manufacturers [25, 12, 38]. As opposed to most current 2D circuits, which use only one active layer[1] for computation, 3-D circuits contain multiple

[1]The active layer is the silicon layer where transistors reside, and metal layers are fabricated above that to connect the transistors together.

active layers which are then interconnected using techniques such as inter-die vias (micron-width wires that are chemically drilled between the layers). In this paper, we argue that an active layer, referred to as a *control plane*, specifically dedicated to security is a potentially cost-effective and computationally efficient methodology for implementing and enforcing a variety of both defensive and offensive security operations.

A control plane provides an opportunity to address security issues in chip multi-processors (CMPs) that share on-chip caches and other resources among multiple processor cores. Multi-core processors are already widely deployed [18], and chip manufacturers have proposed chips with upwards of 100 stripped-down processor cores [22] [29], necessitating the development of techniques to mediate the communication among the cores in an efficient and secure manner. In this paper, we describe a means for cores in the traditional computation plane to communicate with and be observed and controlled by a control plane.

Of course this new technology does not come completely for free since some modifications must be made to the computation plane, and not all designs are amenable to this approach. Specifically, the computation plane must be designed such that the individual elements of the hardware process state of concern are *observable* by the insertion of a *post*, and the default on-chip communication networks can be *overridden*, forcing all security-relevant communication regarding the element up to the control plane so that the desired policies can be enforced. By selectively introducing posts, which provide the control plane direct electronic connections to critical signals on the computation plane, we can create passive monitors on any part of the hardware system state. Active monitoring is enabled through the use of sleep transistors that allow the control plane to turn off portions of the computation plane. Section 3 describes how these posts, sockets, and sleep transistors can be achieved in a minimally intrusive manner. Functions for managing the isolated partitions can then be introduced on the control plane. This makes it possible to isolate computation domains comprised of one or more cores each in the multi-core computation plane and overlay novel security capabilities onto high performance multi-core systems. Specifically, in this paper we make the following contributions:

- We provide analysis showing that 3-D integration, while helpful strictly for performance reasons, offers flexibility for incorporating micro-architectural security mechanisms to enable high-assurance systems. In Section 2, we describe how the control plane can be fused to a multi-core die and how it can be effectively used to enforce a broad class of policies.
- As high-assurance systems represent a small fraction of the total multi-core market, we describe a technique by which the control plane can be integrated in a purely optional and minimally intrusive manner. In Section 3, we show that with minor modifications to the computation plane, we can force critical signals up into the control plane where accesses can be monitored and mediated appropriately.
- Finally, in Section 4 we provide a worked example for how the combination of these two layers (the computation plane and the control plane), working in concert, can help address common security concerns, including the prevention of cache-based side channel attacks.

Before we get into the circuit-level modifications required of the multi-core layer, we begin with a discussion of 3-D integration and the opportunities it presents for high-assurance design.

2 3-D Integration of a Security Layer

In this section we provide background on 3-D interconnect, we describe our threat model and assumptions, and we describe the advantages and architecture of a control plane.

2.1 3-D Chips

The primary goal of this paper is to explore a new method by which defensive and offensive security functionality can be added to a processor. Specifically, we propose a new and modular way to add security

hardware to current and next generation processors through the use of 3-D interconnect. Several 3-D interconnect technologies, such as inter-die vias, are currently being evaluated in industry as a means of stacking multiple chips together. Some potential applications include the stacking of DRAM or bigger caches directly onto the processor die to alleviate memory pressure [37, 43, 26, 45, 21, 17] and designing stacked chips of multiple processors [1]. While the details of this technology are more fully described in Section 3, the main idea is that two pieces of silicon are fused together to form a single chip, and the two active layers of the silicon are connected through inter-die vias (called posts) which run vertically between them. This ability to interconnect multiple active layers means that we can consider optionally adding a layer to a processor specifically for security which would have access to the security dependent signals of the system. A processor with this ability could be sold to customers requiring, for example, high assurance security policy enforcement or other security-specific support, while commodity systems might simply not include this extra control plane.

Large microprocessor manufacturers are unlikely to add dedicated support for high-assurance because this market represents such a small portion of their total customer base. The cost to add functionality directly into a microprocessor is shared by all users, including the vast majority of whom are extremely cost sensitive and do not have high assurance requirements. By fabricating the control plane with functions that are complementary to (but separate from) the main processor, stacked interconnect offers the potential to add security mechanisms on just a small subset of devices without impacting the overall cost of the main processor. Just to be clear, we are advocating the fabrication of a processor which is *always fabricated with connections built in for security*. The difference between the system sold to the cost-sensitive consumer and the one that is sold to the high assurance customer is only whether a specialized security device is actually stacked on top of the standard IC or not.

We must therefore consider the cost of the high assurance system with a control plane stacked on top. There is tangible cost to fabricating systems using 3-D technology as it requires fabricating and testing the security engine, bonding it to its host layer, fabricating the vias necessary for it to communicate with the lower chip, and testing the "joined unit." There is a further cost in terms of the thermal effects. The physical heat sink of the bonded unit attaches to the surface of the host layer as before, but the additional computational density may require the use of more expensive heat sink technology. As this is still an emerging technology, it is difficult to estimate the additional fabrication costs, although many people in the hardware design community are advocating a move towards 3-D interconnect for performance reasons. If this is the case, and 3-D integration becomes mainstream, the incremental cost of adding a layer will be small (especially if one reconfigurable control plane could be used for multiple different families of chip).

The inter-chip 3-D interconnect could take the form of any number of different competing technologies, including chip-bonding, Multi-chip Modules (MCM) [30], chip-stacking with vias [10, 13], or even wireless superconnect [31]. While chip-bonding and MCM technology are already used in a variety of embedded contexts [2, 9], more aggressive interconnect technologies are being heavily researched by several major industrial consortia. Intel, for example, has been investigating 3-D integration to include extra levels of cache. If this technology is included to add extra functionality for consumer machines, it would be only an incremental step to add an additional *optional* control plane.

2.2 Assumptions and Threat Model

The concurrent processing model (e.g., CMP, SMT, SMP) presents significant security problems regarding the separation of activities in the different execution streams. For products based on this model to reach their full potential, these security issues must be overcome. The work described in this paper is intended to contribute to highly effective and efficient solutions for secure processing in concurrent processing environments. We focus on the CMP architecture in order to achieve concrete measurable results, which may be applicable to a wide range of architectures and platforms (e.g., embedded, workstations, and hand held).

In many potential and established applications of CMP, some programs are mutually distrustful, and others must be assumed to be hostile. An example is programs that interface to both the internet and the internal enterprise; another is a program that must cryptographically transform highly valuable data but also interfaces with untrusted software. Therefore, security architectures and mechanisms that can ensure separation of different security domains, with only appropriate interactions allowed between them, may be of great value. CMP and SMP are ideal architectures for providing such separation, but core and processor interference vulnerabilities are inherent in today's designs.

The threats addressed by our work derive from software with unknown behavior (as opposed to programs that are concretely understood), whether that software has been provided by the platform or component manufacturer, or arrives on the platform during runtime. Another concern we address, orthogonal to constraining misbehavior of the software, is the need to passively monitor, or audit, the activities of hardware and software on the computation plane with respect to performance, resource usage, etc.

Outside of the current scope of our work are problems associated with the correctness and integrity of the base hardware (including components on the chip, board resident components and attached devices), which could be caused by design and implementation error or malicious behavior during the hardware lifecycle (e.g., design, fabrication, and integration) or in the field. Active and passive attacks at the hardware level, such as physical removal or probing of the control plane, are also outside of the threat model.

2.3 Advantages of a Control Plane

A control plane provides several practical advantages. First, it provides the opportunity to utilize potentially older, and therefore cheaper, fabrication technology [28] by bonding with different overlays or via reconfigurable logic in the overlay. Furthermore, the security mechanisms are not directly integrated into the computation plane; this makes it more straightforward and less costly to design, build, integrate, and analyze security. Since the security hardware is separate, the additional security enhancements do not consume space or processing cycles in the computation plane.

A security overlay also provides the freedom to place specific security mechanisms directly above where they are needed, without the need for long connections from a security module to the monitored element. For a given device type, reconfiguration of the security policy mechanisms can be implemented, thus efficiently supporting different user requirements.

An overlay also provides several clear theoretical benefits. As always, it is critical to protect security mechanisms, but in this case they may be much less prone to tampering as they are when they are entangled with the monitored design. By definition, a reference monitor must be tamper-resistant, non-bypassable, and simple enough to be formally analyzed [8]. The tamper-resistance comes from the fact that the security mechanisms reside on a physically distinct layer of silicon, and the non-bypassability will be discussed in Section 3. Furthermore, incompleteness results in logic show that while self-monitoring is problematic, monitoring a system externally is more feasible and can be more complete and consistent [32] [15]. With the entire area of the monitoring plane available, security mechanisms can be organized in a way that facilitates their analysis for completeness and correctness. The overlay could also be used to restructure the system into an offensive posture when the system is determined to be under attack.

2.4 Architecture of the Control Plane

While insertion of sophisticated security mechanisms for monitoring or policy enforcement into already complex commodity chips seems unlikely, minimal changes are required to modify their design to accept a 3-D "snap-on" layer. To be commercially feasible, the changes must be unobtrusive and must require a minimum amount of additional logic. Our method requires some small changes to the processor design on the order of a handful of transistors per vertical connection. In addition, the transistors are placed so that the design functions as normal without the control plane.

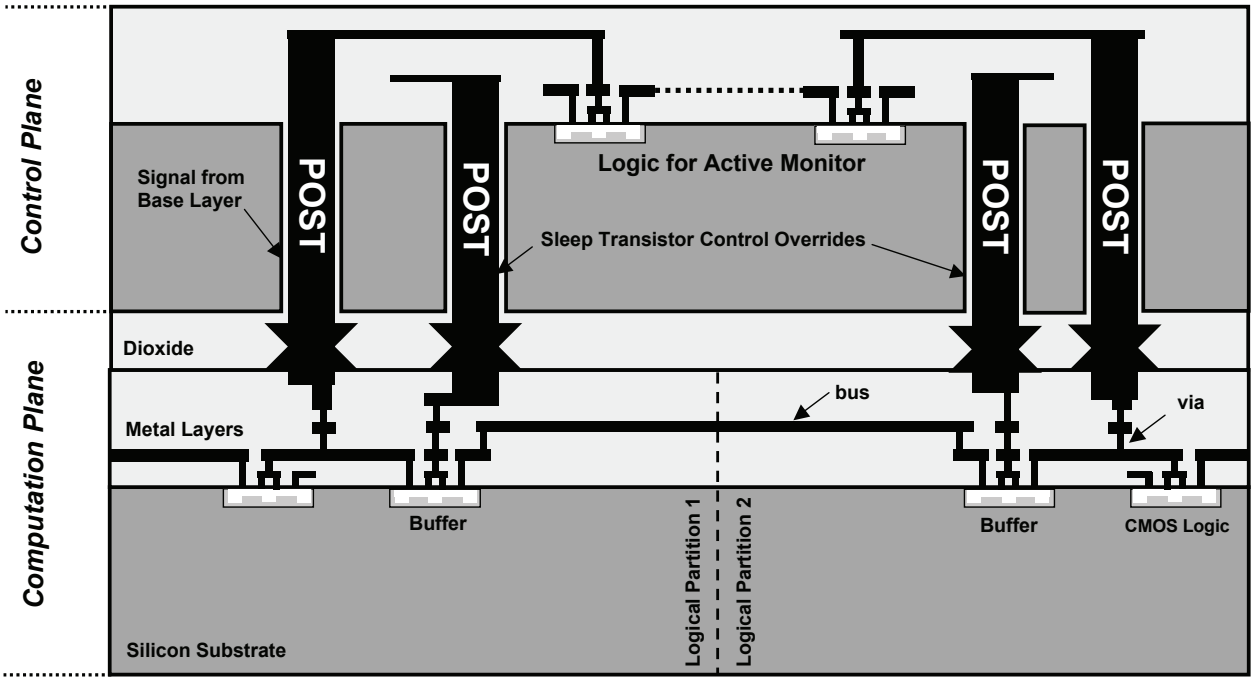

Figure 1: This figure shows four vertical posts connecting the computation plane and the security plane. The two inner posts help to provide the sleep functionality. Signals are rerouted to the control plane through the outer two posts.

A wide range of passive and active security capabilities can be implemented in the control plane. For example, we can build a reference monitor in this control plane to provide memory protection (i.e., a motherboard resource reference validation mechanism, or RVM). We can also implement mechanisms for hardware configuration management, analysis of the computation plane, storage of high integrity code and data (e.g., keys), isolated execution (e.g., proprietary algorithms), tagging, and in selected systems, offensive mechanisms. For example, we can exploit the control plane to tag all traffic traveling over a shared bus according to its source. With our approach, the control plane can manage communication in an efficient and highly scalable manner. The communication overhead of multiple processor cores places a heavy burden on the interconnect because the amount of communication grows in proportion to the number of cores.

While there are many possible applications of a control plane, in this paper we focus on its ability to monitor the computation plane and to override and arbitrate the computation plane, which we discuss in Section 3. We apply these capabilities to the specific problem of cache-based side channel attacks in Section 4. Our solution is to force cache bus traffic to take a "detour" through the control plane, where policy enforcement mechanisms ensure the proper separation of the data.

In systems where resources need to be protected from access by certain users on a persistent basis it is often required for the protection mechanisms to divide resources into groups. that should be treated alike. This division can be achieved through physical or logical separation and results in protection domains (i.e., policy equivalence classes [24]) with which the protection mechanism can efficiently realize the system security policy. In the detour approach, we use physical separation of cores on the computation plane to establish protection domains, and the security mechanisms reside on the control plane.

3 Ramifications to the Computational Layer

The modification to the computation plane to enable the addition of the control plane must be designed so that the bonding operation is not difficult and so the control plane is *fully functional* in its absence. However,

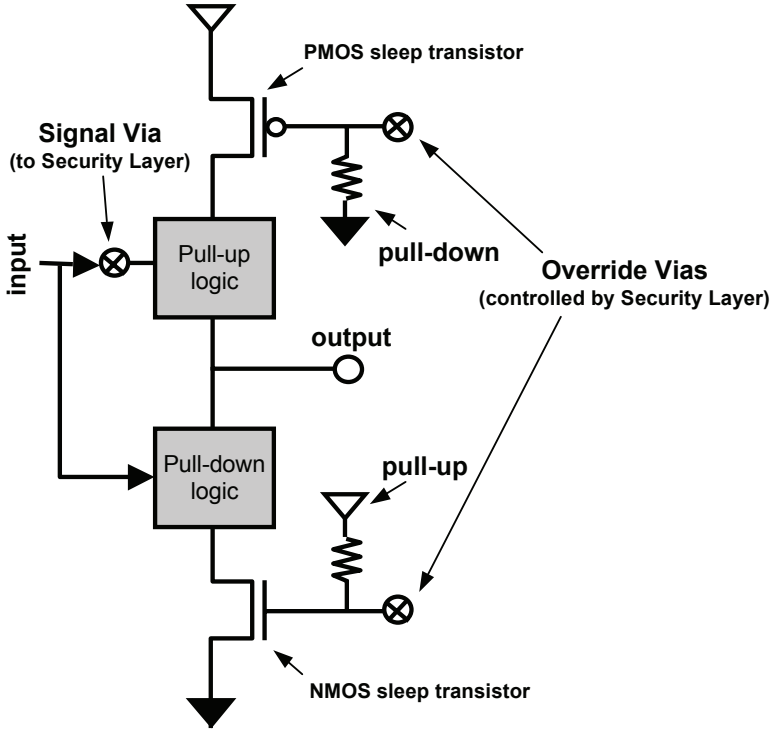

Figure 2: A circuit diagram of the sleep transistors

joined planes will require some shared circuitry in order for the control plane to access the information necessary to perform monitoring and isolation.

3.1 Implementation of Passive Monitors

As discussed in Section 2, one common use of the control plane is simply accessing and analyzing data from the computation plane. For instance, we may wish to monitor accesses to a particular region of memory or audit the use of particular set of instructions. To monitor these events, we must understand when such events are occurring, which necessitates *tapping* some of the wires from the processor. This requires posts and vias to the instruction register and memory wires, and gives us direct access to the currently executing instruction.

This type of passive monitoring is reasonably straightforward to implement in 3-D technology, as it just requires a set of vias to the top of the computation plane, and then a post from there to the control plane. Figure 1 shows this on the left hand side of the figure. The area overhead of this passive style monitoring is analyzed by Mysore et al. [33] in the context of hardware support for debugging. Their conclusion was that, even with very pessimistic assumptions about the technology, there would be less than a 2% increase in the total area on the base level and that there would be no noticeable delay added. The small amount of area overhead is due to the need to save space for the vias across all of the layers of metal. While passive monitoring provides many benefits, the work of Mysore et al. does not consider anything more.

3.2 Implementation of Active Monitors

While passive monitoring allows for auditing, anomaly detection, and the identification of suspicious activities, high assurance systems often require strong guarantees about restrictions to overall system behavior. An active monitor enables control of information flow between cores, the arbitration of communication, and the partitioning of resources.

The key ability needed to support such functionality is to *override* connections in the underlying system. If wires can be overridden by the control plane, then we can force all inter-core communication, memory accesses, and shared signals to travel to the control plane, where they are subject to both examination and control. For instance, we can ensure that confidential data being sent between two cores (which traditionally are forced to traverse a shared bus) is not leaked to an unintended third recipient with access to that bus.

Overriding signals on the computation plane is accomplished in two parts. The first part is to ensure that the monitor has unfettered access to all the signals (tapping), which is, in essence, the same as the passive monitoring scenario described above. The second part is to selectively disable those links, essentially turning off portions of the computation plane (such as a bus). While tapping requires little additional support from the computation plane (essentially just metal vias), some of the active components in the computation plane must be modified to support the highly granular modulation.

The difficulty is that we must remove a functionality (the connection between two components) only by adding a control plane (which cannot physically cut or impede that wire). The computation plane must still be fully functional without the control plane, yet it needs to be constructed so that by wiring in some extra circuitry the targeted computation layer can be completely disabled.

3.3 Efficiently Disabling Wires

The easiest option would simply be to insert a *ground line* onto the wires to be disabled. This has two problems. First, when the wires are driven by a cores (which should be oblivious to the fact that its signals are being routed to the control plane), it is driving current right into ground, creating a short. This in turn consumes a significant amount of power. The second problem is that the wires which are now grounded were the *same ones we were going to tap*, effectively making those taps useless.

An alternate method for disabling links is to disable the portion of the chip responsible for *driving* those links. While this sounds intrusive, we can in fact leverage an existing circuit technique called power gating [39]. Support for power gating is added through the addition of sleep transistors placed between a circuit's logic and its power/ground connections. The sleep transistors act as switches effectively removing the power supply from the circuit. The circuit is awake when the transistors are given the correct signal to be turned on, which provides power to the circuit allowing it to function normally. Alternatively, the sleep transistors can be given the opposite input and turned off, thus disconnecting the power to the circuit, temporarily removing all functionality, and effectively putting the circuit to sleep. Sleep transistors are traditionally used to temporarily disable unused portions of an integrated circuit, saving power by preventing leakage current [40]; however, their use is also beneficial for providing the isolation an active monitor requires.

The sleep transistors can be managed by the control plane by simply providing a post that connects to their gate input. Let us go back to our example of turning off the ability of a component in the computation plane to listen on a bus. To accomplish this, we insert two sleep transistors that can isolate the component from its power supply. When these sleep transistors are activated, the component is temporarily incapacitated. Tri-state buffers arbitrate the use of the bus lines in a circuit. When implementing sleep transistors on tri-state buffers, the capability to isolate the section of our system which uses said tri-state buffers to control bus access is attained.

Adding sleep mode functionality to a circuit requires one PMOS transistor to turn off access to the power supply (Vdd) and one NMOS transistor to cut the connection to ground (see Figure 2). The gate inputs of both of these transistors are routed to the control plane through a post. Furthermore, we must insure that the circuit operates normally (i.e. it is always functional) when the control plane is not included. Therefore, we tie the gate input of the PMOS (NMOS) sleep transistor to ground (Vdd) through a resistor. This insures that the sleep transistor allows access to Vdd (ground), and therefore the circuit is always working as intended. The resistors must be sized appropriately so that when the control plane wishes to turn off the component, it can overpower the constant connection to Vdd (ground), which forces the component to a disabled state.

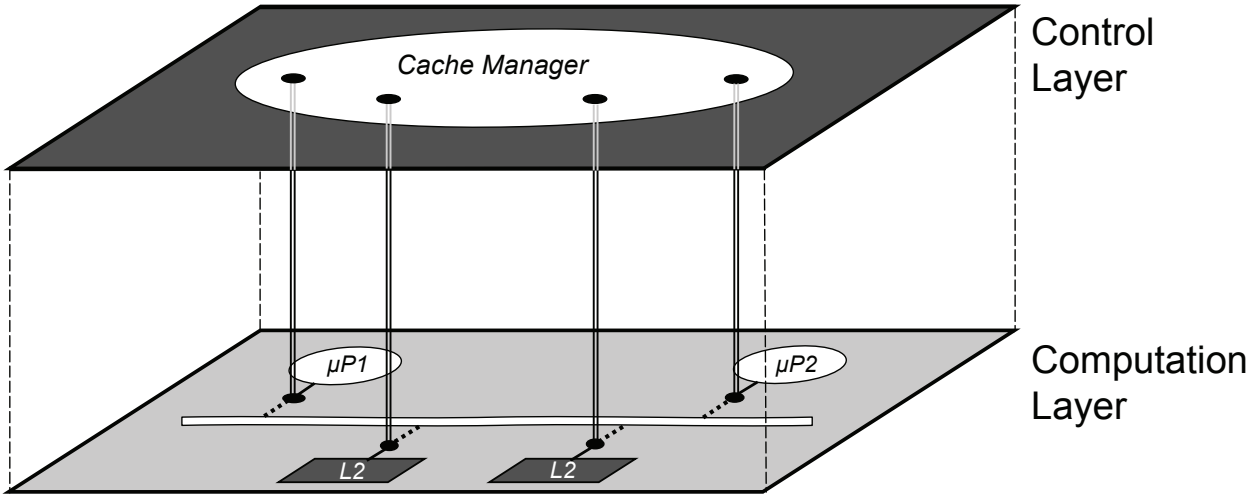

Figure 3: To mitigate the cache-based side channel problem, traffic destined for the cache bus is forced to take a detour to the control plane via the vertical posts. The cache manager in the control plane ensures the separation of each processor's portion of the L2 cache.

The use of sleep mode holds many benefits at a nominal cost. With only a small amount of added hardware (two transistors and two resistors) and posts for connectivity to the control plane, we can selectively turn off portions of the computation plane to force adherence to any specific security policy enforced in the control layer. The exact size of the sleep transistors depends on a variety of factors, which includes the time to turn off/on the circuit and the amount of leakage power savings. These factors are relatively easily varied by changing various physical properties of the sleep transistor, e.g. gate length, oxide thickness and doping [6]. In fact, smaller technology nodes (less than 90 nanometer) need only one sleep transistor due the use of lower power supply voltage [40]. Finally, many modern chips already employ power gating on their shared buses. In this case, the amount of added hardware necessary to apply our security measures is decreased, as only posts to the control plane are needed.

The techniques described in this section provide powerful tools for active and passive monitoring that can vastly improve the number of possible security measures one may enforce on the chip. For example, these techniques can enforce isolation on active components by turning specific communication paths on and off to multiplex access to shared resources. If used appropriately, this can eliminate certain types of side channels by virtualizing the shared resource. The following section illustrates these monitoring techniques to detect and ameliorate cache channel side attacks.

4 Example Application: Cache Side Channels

A chip multi-processor contains multiple general-purpose processor cores on a single die. Preventing processes from "interfering" with each other is an important security requirement. Even if all of the cores and their L1 and L2 caches are logically isolated, lower-level caches such as the L3 cache may be shared. A process running on one core can infer the data such as cryptographic keys belonging to another process by observing either data or instruction cache evictions [4]. The cache therefore is a shared resource that is a vehicle for information flow between cores. Previous work has focused on cache eviction to close the cache-related channels. However, it is expensive to replenish the cache from off-chip memory. Further complicating matters is the fact that multi-core systems may access memory in parallel rather than sequentially. The cache manager on the control plane can implement various techniques to close this channel, such as lattice scheduling with cache eviction if cache access is serialized [16], static partitioning of the cache per

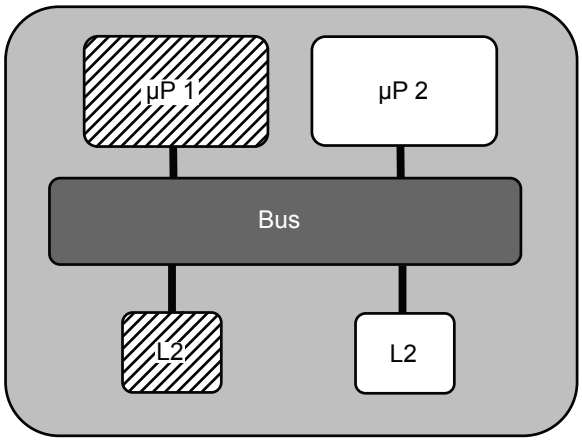

Figure 4: This figure shows a logical view of the computation plane after attaching the control plane. The control plane can prevent $Processor_1$ from accessing $Processor_2$'s portion of the L2 cache and vice-versa.

security domain, and polyinstantiation [27] of cache entries per security domain [44], as we shall see in Section 5.

We propose to use the control plane to prevent the cache from being used as a vehicle for unintended information flows. Figure 3 shows how vertical posts can be used to intercept traffic destined for the cache-bus. In this case, we will add functionality to the control plane to redirect the bus and channel cache traffic to the control plane which will control interaction between the two processors and the two cache segments; all access to the cache will be mediated by the control plane. For example, the cache manager can partition the L2 cache and ensure that each processor can only access its own portion. Figure 4 shows a logical view of the resulting computation plane after attaching the control plane: $Processor_1$ and its portion of the L2 cache are drawn with a hatched pattern, and $Processor_2$ and its portion of the L2 cache are drawn in white. Similarly the cache manager could polyinstantiate cache entries to virtualize the cache.

5 Related Work

5.1 Cache-interference Side Channels

On-chip and board-level resource sharing between cores is often used to enhance CMP performance. However, contention for those resources at the microarchitectural level can provide the basis for "side-channel cryptanalysis" attacks and other covert timing channels. Code and data caches, as well as the branch prediction unit, are some of the shared resources that can be exploited in these attacks. [19, 5, 4] In these cases, one process's use of the resource perturbs the response time of the next processes that accesses it, in a predictable manner. Single-core computers with simultaneous multithreading (SMT), and symmetric multiprocessor (SMP) systems with cache coherency mechanisms, can have similar problems.

Various software and hardware-supported approaches have been reported for preventing cache interference. The primary purpose of secure operating systems is to manage shared resources securely, so it is no surprise that the cache problem has been addressed in software. Cache interference on uni-processors has often been dealt with by "normalizing" the cache (e.g., evicting all cache lines) between execution of different security domains, including mechanisms to avoid doing so unnecessarily. [16] However, this approach is expensive, and is ineffective for processors supporting concurrent execution (CMP, SMT, and SMP), unless access to the cache can be serialized.

One approach to prevent resource contention in a concurrent execution model is to utilize separate physical caches for each core, or provide separate virtual caches within the physical cache (if virtual cache support is available in hardware). [36] [44] Various forms of cache disablement are possible, ranging from

turning it off, to turning it off for certain cores or processes, to turning off the eviction and filling of the cache through use of the processor *no-fill* mode. The latter can be used to create *sensitive sections* [34] of code that could not interfere with the cache behavior observable by other cores or processors – assuming that the code is not interruptible or that the previous processor mode is restored on interrupt, as otherwise, other processes might sense the change to the state of the processor (i.e., to "no-fill"), creating another covert channel. [11]

Specific cryptographic attacks can be defeated or minimized through lowering the bandwidth of the cache channel, such as through nondeterministic ordering of access to cache [35] which makes detailed cache-use profiling difficult; and nondeterministic cache placement [42] [36] or nondeterministic polyinstantiation [14] of cache entries, [44] which, while the specific cause of the interference may be masked, still allows detection of cache misses caused by another process.

The 3-D approach has the advantage of being able to implement many of these schemes for resolving cache contention, while doing it in an isolated environment, without modification to the processor ISA.

5.2 Security in CMPs

There is a great deal of prior work in the area of security for CMPs. Aggarwal et al. have devised a configurable isolation technique for CMPs [7], in which they divide processor resources into isolated domains for improved security, reliability, and performance. A 3-D architecture provides the ability to implement such a configurable isolation strategy with less radical modifications to the commodity processor.

Shi et al. provide security and fault recovery in CMPs with a technique in which one core checks for corruption in the other cores, and the system efficiently recovers compromised cores [41]. Their technique was designed for use in network services, which require high reliability. A 3-D architecture provides the opportunity to implement such a fault recovery technique without the need to sacrifice an entire core on the computation plane.

5.3 Communication in CMPs

Due to the demands placed on the interconnect by multiple cores, the design of an efficient interconnect is *critical* to any CMP, and there is much prior work on communication in CMPs. Abad et al. have proposed a rotary router for a packet-switched network-on-a-chip [3]. Their rotary router uses two independent "rings," and packets can either flow clockwise or counterclockwise. Kim et al. have proposed a flattened butterfly router for high-radix networks-on-a-chip [20]. To address the overhead of complex routers in packet-switched networks-on-a-chip, Kumar et al. have proposed a routing strategy called express virtual channels [23]. By redesigning the routers, their technique improves throughput by bypassing intermediate routers. A 3-D architecture provides an opportunity to further improve the efficiency and security of CMP interconnects.

6 Conclusions and Future Work

Through the techniques in the paper, 3-D integration offers the ability to decouple the development of high assurance security mechanisms from the economics of high performance computing hardware. We provide novel methods by which signals on the computation plane can be overridden by the control plane, enforcing that the monitors on the control plane will be both isolated and non-bypassable. The longer term goal of this work is to help precipitate a renaissance in hardware-assisted security by changing the fundamental trade-offs. We describe various services for the control plane as well as a technique for integrating this plane in a purely optional and minimally intrusive manner to selected points on a commodity integrated circuit. We show how this technique can be applied to address cache-based side channel attacks in chip multi-processors. While the specifics of the example we present shows only how cache side-channel attacks might be mitigated, and certainly there are many other countermeasures and techniques to help deal with this problem, it is the general approach of augmented silicon for security that we believe is our biggest contribu-

tion. Indeed many other hardware-assisted approaches may become feasible through this method including tagging, hardware reference monitors, deep analysis of both the hardware and software of the computation plane, storage assistance for high-integrity code and data, and isolated execution.

References

[1] N. Goldsman A. Akturk and G. Metze. Self-Consistent Modeling of Heating and MOSFET Performance in 3-D Integrated Circuits. *IEEE Transactions on Electron Devices*, 52(11):2395–2403, 2005.

[2] Cristinel Ababei, Yan Feng, Brent Goplen, Hushrav Mogal, Tianpei Zhang, Kia Bazargan, and Sachin Sapatnekar. Placement and Routing in 3D Integrated Circuits. *IEEE Design and Test of Computers*, 22(6):520–531, Nov/Dec 2005.

[3] P. Abad, V. Puente, P. Prieto, and J.A. Gregorio. Rotary router: An efficient architecture for cmp interconnection networks. In *Proceedings of the 34th International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[4] O. Aciíçmez. Yet another microarchitectural attack: Exploiting i-cache. In *Proceedings of the First Computer Security Architecture Workshop (CSAW)*, Fairfax, VA, November 2007.

[5] O. Aciíçmez, J.P. Seifert, and C.K. Koc. Micro-architectural cryptanalysis. *IEEE Security and Privacy Magazine*, 5(4), July-August 2007.

[6] Kanak Agarwal, Harmander Deogun, Dennis Sylvester, and Kevin Nowka. Power gating with multiple sleep modes. *International Symposium on Quality Electronic Design*, 2006.

[7] N. Aggarwal, P. Ranganathan, N.P. Jouppi, and J.E. Smith. Isolation in commodity multicore processors. *Computer*, 40(6), June 2007.

[8] James P. Anderson. Computer security technology planning study. Technical Report ESD-TR-73-51, Air Force Electronic Systems Division, Hanscom AFB, Bedford, MA, 1972. (Also available as Vol. I,DITCAD-758206. Vol. II, DITCAD-772806).

[9] Kaustav Banerjee, Shukri J. Souri, Pawan Kapur, and Krishna C. Saraswat. 3-D ICs: A Novel Chip Design for Improving Deep Submicron Interconnect Performance and Systems-on-Chip Integration. *Proceedings of the IEEE*, 89(5):602–633, May 2001.

[10] Benkart et al. 3D Chip Stack Technology Using Through-Chip Interconnects. *IEEE Design and Test of Computers*, 22(6):512–518, Nov/Dec 2005.

[11] Daniel J. Bernstein. Cache-timing attacks on AES. `http://cr.yp.to/antiforgery/cachetiming-20050414.pdf`, April 2005. Revised version of earlier 2004-11 version.

[12] Bryan Black, Murali Annavaram, Ned Brekelbaum, John DeVale, Lei Jiang, Gabriel H. Loh, Don McCauley, Pat Morrow, Donald W. Nelson, Daniel Pantuso, Paul Reed, Jeff Rupley, Sadasivan Shankar, John Shen, and Clair Webb. Die Stacking (3D) Microarchitecture. *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 469–479, December 2006.

[13] W.R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A.M. Sule, M. Steer, and P.D. Franzon. Demystifying 3D ICs: The Pros and Cons of Going Vertical. *IEEE Design and Test of Computers*, 22(6):498–510, Nov/Dec 2005.

[14] D. E. Denning and T. F. Lunt. A multilevel relational data model. In *Proc. IEEE Symposium on Security and Privacy*, pages 220–234, 1987.

[15] V. Gratzer and D. Naccache. Alien vs. quine. *IEEE Security and Privacy Magazine*, 5(2), March–April 2007.

[16] W.M. Hu. Lattice scheduling and covert channels. In *Proceedings of the 1992 IEEE Symposium on Security and Privacy*, Oakland, CA, May 1992.

[17] Philip Jacob, Okan Erdogan, Aamir Zia, Paul M. Belemjian, Russell P. Kraft, and John F. McDonald. "predicting the performance of a 3D processor-memory chip stack". *IEEE Design and Test of Computers*, 22(6):540–547, Nov/Dec 2005.

[18] J.A. Kahle, M.N. Day, H.P. Hofstee, C.R. Johns, T.R. Maeurer, and D. Shippy. Introduction to the cell multiprocessor. *IBM Journal of Research and Development*, 49(4/5), July/September 2005.

[19] John Kelsey, Bruce Schneier, Chris Hall, and David Wagner. Side channel cryptanalysis of product ciphers. *Journal of Computer Security*, 8(2–3):141–158, 2000.

[20] J. Kim, W.J. Dally, and D. Abts. Flattened butterfly: A cost-efficient topology for high-radix networks. In *Proceedings of the 34th International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[21] Michael B. Kleiner, Stefan A. Kühn, and Werner Weber. Performance Improvement of the Memory Hierarchy of RISC Systems by Applications of 3-D Technology. In *ISCAS*, pages 2305–2308, 1995.

[22] T. Krazit. Intel pledges 80 cores in five years. *CNET News*, 2006.

[23] A. Kumar, L.S. Peh, P. Kudu, and N.K. Jha. Express virtual channels: Towards the ideal interconnection fabric. In *Proceedings of the 34th International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[24] T.E. Levin, C.E. Irvine, C. Weissman, and T.D. Nguyen. Analysis of three multilevel security architectures. In *Proceedings of the First Computer Security Architecture Workshop (CSAW)*, Fairfax, VA, November 2007.

[25] Feihui Li, Chrysostomos Nicopoulos, Thomas Richardson, Yuan Xie, Vijaykrishnan Narayanan, and Mahmut Kandemir. Design and Management of 3D Chip Multiprocessors Using Network-in-Memory. *Proceedings of the 33rd annual International Symposium on Computer Architecture (ISCA)*, pages 130–141, July 2006.

[26] Christianto C. Liu, Ilya Ganusov, Martin Burtscher, and Sandip Tiwari. Bridging the Processor-Memory Performance Gap with 3D IC Technology. *IEEE Design Test*, 22(6):556–564, 2005.

[27] T.F. Lunt, D.E. Denning, R.R. Schell, M. Heckman, and W.R. Shockley. The seaview security model. *IEEE Transactions on Software Engineering*, 16(6), June 1990.

[28] N. Madan and R. Balasubramonian. Leveraging 3D technology for improved reliability. In *Proceedings of the 40th International Symposium on Microarchitecture (MICRO-40)*, Chicago, IL, December 2007.

[29] M. Marshall. Venturebeat: Data center automation firm raises 8m. *San Jose Mercury News*, 2007.

[30] Claude Massit and Nicolas Gerard. Three-dimensional multichip module United State Patents, US 5373189, December 1994.

[31] Miura et al. A 195Gb/s 1.2W 3D-Stacked Inductive Inter-Chip Wireless Superconnect with Transmit Power Control Scheme. In *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, pages 264–265, Feb 2005.

[32] D. Myers. Godel's incompleteness theorem.

[33] S. Mysore, B. Agrawal, S.C. Lin, N. Srivastava, K. Banerjee, and T. Sherwood. Introspective 3-d chips. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, San Jose, CA, October 2006.

[34] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache attacks and countermeasures: the case of AES: (extended version). Technical report, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science,, Rehovot 76100, Israel, October 2005.

[35] D. Page. Theoretical use of cache memory as a cryptanalytic side-channel. Technical Report CSTR-02-003, Department of Computer Science, University of Bristol, June 2002.

[36] D. Page. Partitioned cache architecture as a side channel defence mechanism, 2005.

[37] Kiran Puttaswamy and Gabriel H. Loh. Implementing Caches in a 3D Technology for High Performance Processors. In *IEEE International Conference on Computer Design (ICCD) 2006*, pages 525–532, October 2005.

[38] Kiran Puttaswamy and Gabriel H. Loh. Thermal analysis of a 3D die-stacked high-performance microprocessor. *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, pages 19–24, May 2006.

[39] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand. Leakage current mechanisms and leakage

reduction techniques in deep-submicrometer cmos circuits. *Proceedings of the IEEE*, 91(2), February 2003.

[40] Kaijian Shi and David Howard. Sleep transistor design and implementation simple concepts yet challenges to be optimum. *IEEE VLSI-DAT Taiwan*, 2006.

[41] W. Shi, H.H.S. Lee, L. Falk, and M. Ghosh. An integrated framework for dependable and revivable architectures using multicore processors. In *Proceedings of the 33rd International Symposium on Computer Architecture (ISCA'06)*, Boston, MA, June 2006.

[42] Topham and Gonzalez. Randomized cache placement for eliminating conflicts. *IEEETC: IEEE Transactions on Computers*, 48, 1999.

[43] Yuh-Fang Tsai, Yuan Xie, N. Vijaykrishnan, and Mary Jane Irwin. Three-Dimensional Cache Design Exploration Using 3DCacti. In *IEEE International Conference on Computer Design*. IEEE, October 2005.

[44] Z. Wang and R. Lee. New cache designs for thwarting cache-based side channel attacks. In *Proceedings of the 34th International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[45] Annie Zeng, James Lu, Kenneth Rose, and Ronald J. Gutmann. First-Order Performance Prediction of Cache Memory with Wafer-Level3D Integration. *IEEE Design and Test of Computers*, 22(6):548–555, Nov/Dec 2005.

[46] Inc. Ziptronix. 3D integration for mixed signal applications, 2002.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center — 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library, Code 013 — 1
   Naval Postgraduate School
   Monterey, CA 93943

3. Research Office — 1
   Naval Postgraduate School
   Monterey, CA 93943

4. Theodore Huffmire — 2
   Department of Computer Science
   Naval Postgraduate School
   Monterey, CA 93943

5. Cynthia Irvine — 2
   Department of Computer Science
   Naval Postgraduate School
   Monterey, CA 93943

6. Ryan Kastner — 2
   Department of Computer Science and Engineering
   University of California, San Diego
   9500 Gilman Drive, Mail Code 0404
   La Jolla, CA 92093-0404

7. Timothy Levin — 2
   Department of Computer Science
   Naval Postgraduate School
   Monterey, CA 93943

8. Karl Levitt — 1
   National Science Foundation
   4201 Wilson Blvd.
   Arlington, VA 22230

9. Thuy Nguyen — 2
   Department of Computer Science
   Naval Postgraduate School
   Monterey, CA 93943

10. Timothy Sherwood 2
Department of Computer Science
University of California, Santa Barbara
Office 1119, Harold Frank Hall
Santa Barbara, CA 93106

11. Jonathan Valamehr 2
Harold Frank Hall Room 2152C
University of California
Santa Barbara CA 93106

This page left intentionally blank